

Euclidean Upgrade from a Minimal Number of Segments

Tanja Schilling
Tomáš Pajdla

Center for Machine Perception, Department of Cybernetics
Faculty of Electrical Engineering, Czech Technical University in Prague

Abstract. In this paper, we propose an algebraic approach to upgrade a projective reconstruction to a Euclidean one, and aim at computing the rectifying homography from a minimal number of 9 segments of known length. Constraints are derived from these segments which yield a set of polynomial equations that we solve by means of Gröbner bases. We explain how a solver for such a system of equations can be constructed from simplified template data. Moreover, we present experiments that demonstrate that the given problem can be solved in this way.

1 Introduction

A projective reconstruction can be computed from image correspondences alone, without any information about calibration or pose of the cameras. But additional knowledge about the scene is required to subsequently recover Euclidean structure [7]. The problem of computing such a homography that upgrades a projective reconstruction to a metric one has been treated in many publications, e.g. in [12, 8].

In our case, the additional knowledge about the scene comprises a set of segments with known lengths, i.e. pairs of points with known distances between them. This problem has already been addressed in [11], where parallel projection was assumed and hence an affine reconstruction was supposed to be known.

The projective case was considered in [13]. The authors introduced the concept of a Quadric of Segments (QoS), defined in a higher-dimensional space by the set of segments of known length, that can be computed linearly. Euclidean structure can be recovered from the QoS in closed form, exploiting the fact that all spheres intersect the plane at infinity in the absolute conic [7]. Although the linear computability is certainly advantageous, the high number of segments (54 in the 3D case) required to define the QoS constitutes a main drawback. Given that not many segments of known length are available in various conceivable settings, we therefore seek a solution that requires only as few segments as possible.

The constraints on the upgrading homography derived from these segments constitute a system of non-linear algebraic equations. Polynomial systems occur in various computer vision problems and many of them have been solved by means of Gröbner bases [4, 5, 15]. But there is no easy, straightforward method to solve general polynomial systems efficiently and robustly. Instead, each particular problem usually requires the manual design of a suitable Gröbner basis solver. However, an automatic generator of minimal problem solvers was presented in [10].

In this paper, we show that it is possible to compute the homography that upgrades a projective reconstruction to a Euclidean one algebraically from only 9 segments of known length. The following section states the problem in detail. Section 3 briefly introduces the notion of Gröbner bases, on which our approach is based. We explain how a special solver for the problem at hand can be constructed and applied in section 4. Finally, experimental results are presented in section 5.

2 Problem statement

The image projection of a scene point X_i , represented by its homogeneous coordinates [7], is denoted as $x_i \propto \mathbf{P}X_i$, i.e. an $\alpha_i \in \mathbb{R}$ exists, such that $x_i = \alpha_i \mathbf{P}X_i$. Let us assume that we have measured n image coordinates $\hat{x}_i \not\propto x_i, i \in \mathbb{N}$, and computed a projective reconstruction $(\hat{\mathbf{P}}, \{\hat{X}_i\})$, where $\hat{\mathbf{P}} \not\propto \mathbf{P}$ and $\hat{X}_i \not\propto X_i$, from those points, such that $\hat{x}_i \propto \hat{\mathbf{P}}\hat{X}_i$. Aiming at recovering Euclidean structure from this, we have to find a non-singular matrix $\mathbf{H} \in \mathbb{R}^{4 \times 4}$ which upgrades \hat{X}_i and $\hat{\mathbf{P}}$ by a projective transformation, such that $\mathbf{P}X_i \propto \hat{\mathbf{P}}\mathbf{H}^{-1}\mathbf{H}\hat{X}_i$ and $X_i \propto \mathbf{H}\hat{X}_i$.

In order to reduce the number of unknowns in \mathbf{H} , we fix the reference frame by choosing three points \hat{X}_i that determine the origin, the x -axis and the xy -plane. From now on, let us assume that all points \hat{X}_i have already been mapped by a suitable similarity transform such that there exist three points $(0, 0, 0, 1)^\top, (\hat{x}_i, 0, 0, 1)^\top$ and $(\hat{x}_j, \hat{y}_j, 0, 1)^\top, \hat{x}_i, \hat{x}_j, \hat{y}_j \in \mathbb{R} \setminus \{0\}$, which determine the coordinate frame.

Mapping the point of origin to itself, $(0, 0, 0, 1)^\top \propto \mathbf{H}(0, 0, 0, 1)^\top$, points on the x -axis to the x -axis, $(x_i, 0, 0, 1)^\top \propto \mathbf{H}(\hat{x}_i, 0, 0, 1)^\top$, and points in the xy -plane to the xy -plane again, $(x_j, y_j, 0, 1)^\top \propto \mathbf{H}(\hat{x}_j, \hat{y}_j, 0, 1)^\top$, the transformation matrix has to be of the form

$$\mathbf{H} = \begin{pmatrix} h_1 & h_2 & h_3 & 0 \\ 0 & h_4 & h_5 & 0 \\ 0 & 0 & h_6 & 0 \\ h_1 - h_9 & h_7 & h_8 & h_9 \end{pmatrix}. \quad (1)$$

Two constraints on \mathbf{H} easily arise from that. First, the projection matrix has to be invertible and therefore \mathbf{H} must fulfill

$$0 \neq \det(\mathbf{H}) = h_1 h_4 h_6 h_9. \quad (2)$$

Secondly, we fix the scale of \mathbf{H} by demanding that for one point $\hat{X}_i, 1 \leq i \leq n$

$$1 = \mathbf{H}_4 \hat{X}_i = X_{4i} \quad (3)$$

with \mathbf{H}_4 denoting the 4-th row of \mathbf{H} and X_{4i} being the 4-th element of X_i .

Furthermore, we assume that there are N pairs of points, (X_i, Y_i) which represent segments of known lengths d_i , such that $\|X_i - Y_i\| = d_i$ for all $i = 1 \dots N$. Replacing X_i by $\mathbf{H}\hat{X}_i$, yields the following constraint on \mathbf{H} :

$$f_i(\mathbf{h}) = 0 = \sum_{l=1}^3 (\mathbf{H}_l \hat{X}_i \mathbf{H}_4 \hat{Y}_i - \mathbf{H}_4 \hat{X}_i \mathbf{H}_l \hat{Y}_i)^2 - (\mathbf{H}_4 \hat{X}_i \mathbf{H}_4 \hat{Y}_i)^2 d_i^2 \quad (4)$$

where \mathbf{H}_l denotes the l -th row of \mathbf{H} .

Equation (4) constitutes a homogeneous polynomial of degree 4 in 9 variables, with 97 terms in the general case. Therefore, at least 9 segments (X_i, Y_i) are required to obtain the 9 equations, which determine \mathbf{H} . Introducing an additional variable h_{10} , we can rewrite the inequality (2) as equality [2]

$$0 = 1 - h_1 h_4 h_6 h_9 h_{10}, \quad (5)$$

as well as

$$0 = 1 - \mathbf{H}_4 \hat{X}_i. \quad (6)$$

Now, the problem is to solve the system of $m = N + 2$ algebraic equations, $N \geq 9$, in 10 variables h_1, \dots, h_{10} ,

$$0 = f_1(\mathbf{h}) = \dots = f_m(\mathbf{h}). \quad (7)$$

3 Gröbner bases

Systems of polynomial equations can be solved efficiently by means of Gröbner bases [6]. F denotes the set of m polynomials $F = \{f_1(\mathbf{h}), \dots, f_m(\mathbf{h}) | f_i(\mathbf{h}) \in K[h_1, \dots, h_n]\}$ in n variables $\mathbf{h} = (h_1, \dots, h_n)$ over a field K . The ideal $I = \langle F \rangle$ generated by F is the set of all polynomial linear combinations

$$I = \left\{ \sum_{i=1}^m f_i(\mathbf{h}) q_i(\mathbf{h}) | q_i(\mathbf{h}) \in K[h_1, \dots, h_n] \right\}. \quad (8)$$

A Gröbner basis is a special set of generators with desirable algorithmic properties. In particular, a Gröbner basis of an ideal I has the same set of solutions as I . But similar to a system of linear equations after Gaussian elimination, the solutions of I can be easily identified in the corresponding Gröbner basis w.r.t. a lexicographical monomial ordering [6].

Theoretically, the Gröbner basis can be computed from any generating set of I by a method called Buchberger's algorithm [6]. The basic mechanism is to take each pair $(f_i(\mathbf{h}), f_j(\mathbf{h}))$ from F , $f_i(\mathbf{h}) \neq f_j(\mathbf{h})$, compute its S -polynomial (see appendix), reduce it by F and add the remainder to F if it is not zero. This is done until the S -polynomials of all pairs in F reduce to zero.

This problem is known to be EXPSPACE-complete in general [9]. Nevertheless, much better bounds can be found for many cases that actually occur in practice and several well-known methods exist to improve the basic algorithm.

However, computing a Gröbner basis straightaway from the set of equations introduced in section 2 with floating point arithmetic is not practicable for two reasons: One obstacle is that even with improved versions of Buchberger's algorithm many S -polynomials are constructed in vain as they finally reduce to zero and hence do not contribute to the final basis, merely slowing the entire Gröbner basis computation down. Another difficulty arises as a result of accumulating round-off errors in floating point arithmetic during repeated reductions of S -polynomials. Except for extremely simple examples, these round-off errors make it impossible to decide whether a particular coefficient very close to zero should be considered as zero causing the cancellation of the corresponding term or not. Thus a special solver that is adapted to the given problem has to be created.

4 Solver design and application

We build our solver by computing a Gröbner basis of a template system of equations first. These polynomials are generated as explained in section 2, but originate from a simplified set of segments (X_i, Y_i) with integer coordinates. Calculations are done in a finite field \mathbb{Z}_p . During the computation, we record which of the pairs of polynomials taken from the intermediate basis form S -polynomials that do not reduce to zero during the process. In so doing, we get a computation template that contains only those pairs of polynomials that actually contribute to the final Gröbner basis. In principle, this step has to be performed only once.

Afterwards, we can apply that solver to compute a Gröbner basis from a general polynomial system in floating point arithmetic, taking only the previously recorded pairs of polynomials into account, which speeds up this procedure. To avoid the above mentioned problematic effects of round-off errors, we could have memorized when which integer coefficient during the template computation gets zero as proposed in [17] and proceed accordingly. But since storing and reading this information takes a considerable amount of memory and time, we favor the more practicable way of simply

processing the template data once again, simultaneously with the polynomial system that we want to solve. Whenever a coefficient of the template data (computed in \mathbb{Z}_p) becomes zero, the corresponding floating point coefficient is set to zero, too.

We assume that the sequence of operations to construct the Gröbner basis is basically identical for different sets of polynomials, given that the equations in those sets contain the same monomials and differ only in their coefficients [16]. With this assumption, we rely on the fact that Buchberger's algorithm and its improved variants do not consider the values of non-zero coefficients for the choice of critical pairs, the detection of unnecessary pairs or the selection of reducers.

The crucial point in this scheme is to find an appropriate template system that is simple enough to be feasible yet general enough to be used for the original problem. The difficulty is in the fact that having identical monomials in the template polynomials and in the original is required to achieve the same sequence of computation but does not necessarily lead to the desired result.

To generate the template set, we simplified the general problem by using small integers coordinates for all segments $\|X_i - Y_i\| = d_i$, such that also $d_i \in \mathbb{Z} \setminus \{0\}$. That means each segment has to fulfill

$$d^2 = \|X_i - Y_i\|^2 = a^2 + b^2 + c^2, \quad (9)$$

where $a, b, c, d \in \mathbb{Z}$, $d \neq 0$, thus forming a Pythagorean quadruple if $a, b, c \in \mathbb{Z} \setminus \{0\}$ or a Pythagorean triple respectively if $a = 0$, $b \neq 0$ and $c \neq 0$. The possibility to use such a special polynomial system as a template for the general problem is justified by the fact that every triple in \mathbb{R}^3 has a sufficiently precise scaled representation as a Pythagorean triple in \mathbb{Z}^3 [14]. Hence, we are looking for a generic Pythagorean case which is feasible to compute and at the same time implementable for a wide range of practically occurring systems originating from real coefficients.

5 Experiments

5.1 Solver generation

Although we aim to generate a solver to compute \mathbf{H} from the minimal number of segments $N = 9$, we conducted experiments to assess the influence of the number of segments to the presented method.

The template data set from which the solver is built consists of N pairs of points (X_i, Y_i) that are generated from Pythagorean quadruples as outlined above, and a homography \mathbf{H} . More precisely, $N - 1$ Pythagorean quadruples (a_i, b_i, c_i, d_i) , where $0 < a_i, b_i, c_i, d_i < 50$, are selected randomly, such that $(a_i, b_i, c_i, d_i) \neq (sa_j, sb_j, sc_j, sd_j)$ for any scaling factor $s \in \mathbb{Z}$, for all $i, j = 1 \dots N$ and $i \neq j$. Euclidean coordinates of X_i are chosen randomly as well, whereupon the points Y_i are calculated, such that

$$X_i - Y_i = (X_{1i}, X_{2i}, X_{3i}, 1)^\top - (Y_{1i}, Y_{2i}, Y_{3i}, 1)^\top = (a_i, b_i, c_i, 0)^\top. \quad (10)$$

In general, X_i and Y_i coordinates are non-zero integers, $0 < |X_{ji}|, |Y_{ji}| < 100$, except for the first 2 segments

$$\begin{aligned} X_1 &= (0, 0, 0, 1)^\top, & Y_1 &= (Y_{11}, 0, 0, 1)^\top, \\ X_2 &= (X_{12}, X_{22}, 0, 1)^\top, & Y_2 &= (Y_{12}, Y_{22}, Y_{32}, 1)^\top, \end{aligned} \quad (11)$$

which are required to determine the x -axis and the xy -plane as explained in section 2. \mathbf{H} is generated according to equation (1), with random non-zero integer elements $0 < h_k < 20$ for all $k = 1 \dots 9$.

The resulting N pairs of distorted points

$$(\mathbf{H}^{-1}X_i, \mathbf{H}^{-1}Y_i) = (\hat{X}_i, \hat{Y}_i) \quad (12)$$

yield $N + 2$ polynomials according to equations (4), (5) and (6). A Gröbner basis is computed from this system of equations in a finite field \mathbb{Z}_p with $p = 332251314113$ as this proved to be a sufficiently large prime number in earlier experiments. In that way, solvers are built for 20 different template data sets for each of 10 distinct values of N .

As outlined in section 3, the intermediate basis F grows until the S -polynomials of all pairs $(f_i(\mathbf{h}), f_j(\mathbf{h}))$ in F reduce to zero, usually producing a large final Gröbner basis in this way. Therefore, its reduced Gröbner basis [6] is computed afterwards, from which the solution can be easily obtained and which is unique for the considered ideal and a given monomial ordering.

The reduced Gröbner basis for the ideal generated by the considered polynomial system derived from N segments consists of the following 13 simple equations g_i , that only differ in the coefficients c_{ij} .

$$\begin{aligned} g_1 = 0 &= c_{1,1}h_1 + c_{1,2} & g_2 = 0 &= c_{2,1}h_2 + c_{2,2} \\ g_3 = 0 &= c_{3,1}h_3 + c_{3,2} & g_4 = 0 &= c_{4,1}h_7 + c_{4,2} \\ g_5 = 0 &= c_{5,1}h_8 + c_{5,2} & g_6 = 0 &= c_{6,1}h_9 + c_{6,2} \\ g_7 = 0 &= c_{7,1}h_4 + c_{7,2}h_5 & g_8 = 0 &= c_{8,1}h_5^2 + c_{8,2} \\ g_9 = 0 &= c_{9,1}h_6^2 + c_{9,2} & g_{10} = 0 &= c_{10,1}h_{10}^2 + c_{10,2} \\ g_{11} = 0 &= c_{11,1}h_5h_6 + c_{11,2}h_{10} & g_{12} = 0 &= c_{12,1}h_5h_{10} + c_{12,2}h_6 \\ g_{13} = 0 &= c_{13,1}h_6h_{10} + c_{13,2}h_5 \end{aligned} \quad (13)$$

These equations yield 4 solutions for \mathbf{H} , varying in the signs of h_4 , h_5 , and h_6 , that maps all points \hat{X}_i either to

$$\begin{aligned} X_i^{(++)} &= (X_{1i}, X_{2i}, X_{3i}, 1)^\top, & X_i^{(-+)} &= (X_{1i}, -X_{2i}, X_{3i}, 1)^\top, \\ X_i^{(+-)} &= (X_{1i}, X_{2i}, -X_{3i}, 1)^\top \text{ or } & X_i^{(---)} &= (X_{1i}, -X_{2i}, -X_{3i}, 1)^\top, \end{aligned} \quad (14)$$

and \hat{Y}_i analogously, depending on whether both h_5 and h_6 are positive, $h_5 < 0$ and $h_6 > 0$, $h_5 > 0$ and $h_6 < 0$ or both are negative. Hence, a 180° rotation of all segments $(X_i^{(++)}, Y_i^{(++)})$ about the x -axis corresponds to the set of segments $\{(X_i^{(---)}, Y_i^{(---)})\}$, and mirroring these sets on the xz -plane yields the other two sets of segments $\{(X_i^{(-+)}, Y_i^{(-+)})\}$ and $\{(X_i^{(+-)}, Y_i^{(+-)})\}$.

In contrast to the explained basic Buchberger's algorithm, sets of pairs of polynomials $\{(f_i(\mathbf{h}), f_j(\mathbf{h}))\}$ instead of single pairs are reduced simultaneously during so called multi-reduction steps in the modified version [3] of the algorithm that we use in our experiments.

The so computed reduced Gröbner bases from various template data sets and different values of N vary only in the coefficients c_{ij} of equations (13). Table 1 shows that the Gröbner basis computation is faster for higher number of segments in terms of required multi-reduction steps and computing time. But more than 50 segments do not speed it up further. Unsurprisingly, the final basis contains more equations if more multi-reduction steps were necessary to compute it.

Next, the generated solvers are tested to find the minimally required precision for floating point arithmetic. More exactly, we apply each solver to its template data set, but compute its reduced Gröbner basis in floating point arithmetic this time as explained in section 4. If the absolute values of the difference between the so computed segment

N	Basis size	Multi-reductions	Time in msec
9	500	28	5170
10	424	24	3828
12	364	19	759
15	206	17	140
20	105	17	11
25	95	14	10
50	67	12	5
100	67	12	7
200	67	13	20
500	67	16	57

Table 1: Solver generation: Size of the computed (not yet reduced) Gröbner basis, required multi-reduction steps and computation time for different numbers of segments N .

N	Precision in bit	Time (min:sec.msec)
9	1088	2:49.23
10	512	1:24.16
12	448	0:12.50
15	384	0:02.59
20	192	0:00.11
25	256	0:00.10
50	256	0:00.05
100	256	0:00.07
200	256	0:00.20
500	256	0:00.57

Table 2: Solver testing: Required floating point precision and computation time to calculate the correct solution for different numbers of segments N .

length and the true segment length is smaller than 10^{-9} , computed and true segment length are considered to be equal and the used precision is therefore sufficient. Our implementations make use of the GNU Multiple Precision Arithmetic Library [1] to handle high precision floating point arithmetic as well as large integers.

The required precisions for different N are given in table 2. As expected, a higher precision is required for a lower number of segments, but using more than 50 segments does not decrease the necessary precision further.

5.2 Application to exact data

Now, we apply the solver to various floating point data sets. Each data set comprises N pairs (X_i, Y_i) , and a homography \mathbf{H} . The Euclidean coordinates of the points X_i and Y_i are in general randomly chosen within a cube of side lengths 10. All coordinates $X_{ji}, Y_{ji} \in \mathbb{R}$ are chosen such that $10X_{ji}, 10Y_{ji} \in \mathbb{Z}$, i.e. only coordinates with at most one digit after the decimal point are considered. Furthermore, $0 \leq X_{ji}, Y_{ji} \leq 10$ for all $j = 1 \dots 3$ and $X_{4i} = Y_{ji} = 1$ for all $i = 1 \dots N$. The true segment length $d_i = \|X_i - Y_i\|$ varies.

\mathbf{H} is chosen in a way that \mathbf{H}^{-1} maps 3 vertices V_l of the cube to $V_l + T_l \propto \mathbf{H}^{-1}V_l$, where T is a random vector of independent zero-mean Gaussian components with typical deviation 1, and all $h_k \in \mathbb{R}$. The system of polynomial equation is obtained from N pairs $(\hat{X}_i = \mathbf{H}^{-1}X_i, \hat{Y}_i = \mathbf{H}^{-1}Y_i)$ according to equations (4), (5) and (6). For different values of N , each of the 20 solvers created in section 5.1 is applied to 20 different data sets, to compute the respective reduced Gröbner bases, and then the sought homographies \mathbf{H}' . Subsequently, the distorted points \hat{X}_i and \hat{Y}_i are upgraded to $X'_i = \mathbf{H}'\hat{X}_i$ and $Y'_i = \mathbf{H}'\hat{Y}_i$ and the segment lengths $d'_i = \|X'_i - Y'_i\|$ are computed.

The error is calculated as the standard deviation of the differences between the computed and true segment lengths divided by the average segment length $\sigma(d_i - d'_i)/\mu(d_i)$. However, solvers may fail under certain conditions, such that no solution can be obtained. In table 3, the percentage of successful computed solutions and the mean errors of those solutions are summarized for different numbers of segments N .

In our experiments, the solvers failed for instance on 2 data sets due to the chosen \mathbf{H}^{-1} and a pair (X_i, Y_i) , that caused one or more coefficients in the resulting polynomial to be zero. Hence, a floating point exception occurred when division by zero was attempted during reduction by this polynomial. However, this problem occurred only in less than 0.1% of all polynomials that we generated from test data. In all cases,

N	Success rate	Mean error
9	100%	7.5e-37
10	100%	3.4e-45
12	100%	4.1e-70
15	100%	3.4e-82
20	100%	2.3e-129
25	100%	1.4e-137
50	95%	2.3e-139
100	95%	1.2e-141
200	85%	4.5e-144
500	85%	3.2e-149

Table 3: Solver application to exact data: Success rate and mean errors for different numbers of segments N .

N	Success rate	Mean error
25	4%	0.26
50	48%	0.11
100	49%	0.11

Table 4: Solver application to noisy data, $\sigma = 0.001$: Success rate and mean errors for different numbers of segments N .

	20 different data sets	Σ
19 different solvers	××××××××××××××××××	9
	××√×××××√√×××××××	9
	×××××××√√√×××××××	9
	××√×××××√√√××××××	12
	××√×××××√√√××××××	9
	××√×××××√√√××××××	9
	××√×××××√√√××××××	8
	××√×××××√√√×××××√×	12
	××××××××√√√××××××√	9
	××××××××√√√×××××××	9
	××××××××√√√×××××√×	9
	××√×××××√√√×××××√×	10
	√×××××××√√√×××××√×	11
	××××××××√√√×××××√×	9
	××××××××√√√×××××××	9
	××√×××××√√√×××××××	10
	××××××××√√√×××××√×	9
	××√×××××√√√××××××√	11
	××√×××××√√√×××××√×	12

Table 5: Solver application to noisy data, $N = 50$, $\sigma = 0.001$: Success (✓) and failures (✗) for different applying each of 19 different solvers to each of 20 distinct data sets. The last column displays the number of successful computations per solver.

where the solver could be successfully applied to compute a solution, the upgraded segment lengths nearly equal the true length.

5.3 Application to noisy data

Finally, we investigated the effect of noise in the data. We used the same data sets as above, but considered only those solvers and datasets for which the computation did not fail in the previous experiment. A random vector of independent zero-mean Gaussian components was added to each \hat{X}_i and \hat{Y}_i , and the polynomial system was derived from this noisy data.

Our experiments revealed that even for a very small noise standard deviation, the computation completely failed in many cases. Here, failing means that though a reduced Gröbner basis was computed, no real valued solution could be obtained from that because the reduced basis contained equations like $0 = c_1 h_k^2 + c_2$ with positive coefficients c_1 and c_2 . Selected results are illustrated in table 4. For larger numbers of segments the failure percentage as well as the mean error decreases, such that on some data sets none of the generated solver fails for $N \geq 50$ as illustrated in table 5.

6 Conclusion

In this paper, we proposed an algebraic way to compute a homography to upgrade a preliminary projective reconstruction to an Euclidean one by means of constraints derived from a minimal number of segments with known lengths. We believe, this could be useful in environments where prior camera calibration is impracticable but a small number of distances between points is known.

We have shown that it is possible to solve this problem using only 9 segments and demonstrated how a corresponding solver can be constructed from simplified template data. However, our experiments revealed that the presented method is very likely to fail on noisy data. This has to be investigated more thoroughly in order to be able to apply our technique to real world data.

Acknowledgements

This work has been supported by the project PRoViDE EU FP7-SPACE-312377 and FP7-SME-2011-285839 De-Montes.

Appendix: Notation

We use the notations *term* and *monomial* as they are explained in [6], i.e. given a polynomial ring $K[x_1, x_2, \dots, x_n]$, a monomial is a product of the form

$$x_1^{\alpha_1} \cdot x_2^{\alpha_2} \cdots x_n^{\alpha_n} = x^\alpha,$$

with non-negative integer exponents $\alpha_1, \alpha_2, \dots, \alpha_n$. A term then denotes the product $a_\alpha x^\alpha$ of a monomial and a non-zero coefficient $a_\alpha \in K$.

An *S-polynomial* of a pair of polynomials (f, g) is computed as

$$S(f, g) = x^\gamma (\text{LT}(f))^{-1} f - x^\gamma (\text{LT}(g))^{-1} g$$

where $x^\gamma = \text{LCM}(\text{LM}(f), \text{LM}(g))$ is the least common multiple of $\text{LM}(f)$ and $\text{LM}(g)$. $\text{LM}(f)$ denotes the leading monomial and $\text{LT}(f)$ the leading term of f w.r.t. a monomial ordering.

References

- [1] GMP: The GNU Multiple Precision Arithmetic Library. <http://gmplib.org>.
- [2] Thomas Becker and Volker Weispfenning. *Gröbner Bases: A Computational Approach to Commutative Algebra*. Graduate Texts in Mathematics. Springer, New York, Heidelberg, 1993.
- [3] Michael Brickenstein. Slimgb: Gröbner Bases with Slim Polynomials. Reports on Computer Algebra 35, Centre for Computer Algebra, University of Kaiserslautern, 2005.
- [4] Martin Bujnak, Zuzana Kukelova, and Tomas Pajdla. 3D Reconstruction from Image Collections with a Single Known Focal Length. In *Proceedings of the 12th International Conference on Computer Vision*, 2009.
- [5] Martin Byröd, Klas Josephson, and Kalle Aström. Fast Optimal Three View Triangulation. In *Proceedings of the 8th Asian Conference on Computer Vision*, 2007.
- [6] David A. Cox, John Little, and Donal O'Shea. *Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra*. Undergraduate Texts in Mathematics. Springer, New York, Heidelberg, 1992.

- [7] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, Cambridge, 2006.
- [8] Anders Heyden and Kalle Astrom. Euclidean Reconstruction from Image Sequences with Varying and Unknown Focal Length and Principal Point. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1997.
- [9] Klaus Kühnle and Ernst W. Mayr. Exponential Space Computation of Gröbner Bases. In *Proceedings of the 1996 International Symposium on Symbolic and Algebraic Computation*, 1996.
- [10] Zuzana Kukelova, Martin Bujnak, and Tomas Pajdla. Automatic Generator of Minimal Problem Solvers. In *Proceedings of the 10th European Conference on Computer Vision*, 2008.
- [11] David Liebowitz and Stefan Carlsson. Uncalibrated Motion Capture Exploiting Articulated Structure Constraints. *International Journal of Computer Vision*, 51(3):171–187, 2003.
- [12] Jean Ponce. On Computing Metric Upgrades of Projective Reconstructions under the Rectangular Pixel Assumption. In Marc Pollefeys, Luc Van Gool, Andrew Zisserman, and Andrew Fitzgibbon, editors, *Second European Workshop on 3D Structure from Multiple Images of Large-Scale Environments*, Lecture Notes in Computer Science, pages 52–67. Springer Berlin Heidelberg, 2001.
- [13] José I. Ronda and Antonio Valdes. Euclidean Upgrading from Segment Lengths. *International Journal of Computer Vision*, 90(3):350–368, July 2010.
- [14] P. Shiu. The Shapes and Sizes of Pythagorean Triangles. *The Mathematical Gazette*, 67(439):33, March 1983.
- [15] H. Stewenius, D. Nister, F. Kahl, and F. Schaffalitzky. A Minimal Solution for Relative Pose with Unknown Focal Length. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2005.
- [16] Carlo Traverso. Gröbner trace algorithms. In *Proceedings of the 1988 International Symposium on Symbolic and Algebraic Computation*, pages 125–138, 1988.
- [17] Carlo Traverso and Alberto Zanzi. Numerical stability and stabilization of groebner basis computation. In *Proceedings of the 2002 International Symposium on Symbolic and Algebraic Computation*, ISSAC '02, pages 262–269, New York, NY, USA, 2002. ACM.